

PAT-NO: JP401102644A

DOCUMENT-IDENTIFIER: JP 01102644 A

TITLE: PIPELINE TYPE PROCESSING UNIT

PUBN-DATE: April 20, 1989

INVENTOR-INFORMATION:

NAME

SCHWARZ, ERIC M

VASSILIADIS, STAMATIS

ASSIGNEE-INFORMATION:

NAME

COUNTRY

INTERNATL BUSINESS MACH CORP <IBM> N/A

APPL-NO: JP63233855

APPL-DATE: September 20, 1988

INT-CL (IPC): G06F009/38, G06F015/16

ABSTRACT:

PURPOSE: To introduce a dynamic MIMD pipeline to a computer system by reading a succeeding instruction stream even when pipeline processing and execution of a 1st instruction stream are not finished in a pipeline circuit.

CONSTITUTION: The processing unit includes receiving means 21, 22, 25 receiving an input instruction, plural pipeline processing means 26 executing an input instruction, and a dynamic activity recording table 27 to control the instruction execution of the processing means 26. Each of processing means 26a-26d stores and executes plural instructions and they are identified by definite identifiers (pipe numbers) 1-4 respectively. Since the instructions stored in each of the processing means 26a-26d are in various execution stages,

a table 27 stores information relating to the instructions stored in each of the processing means 26a-26d. The information includes an identifier of the processing means to execute the instruction. Thus, the dynamic plural-instruction stream plural-data (MIMD) pipeline is used for the computer system to attain a high efficiency.

COPYRIGHT: (C)1989,JPO

⑫ 公開特許公報(A)

平1-102644

⑤ Int. Cl.⁴G 06 F 9/38
15/16

識別記号

3 5 0
3 7 0

庁内整理番号

A-7361-5B
6745-5B

④ 公開 平成1年(1989)4月20日

審査請求 有 請求項の数 1 (全21頁)

⑬ 発明の名称 バイブライン式処理装置

⑭ 特 願 昭63-233855

⑮ 出 願 昭63(1988)9月20日

優先権主張 ⑯ 1987年9月30日 ⑰ 米国(US) ⑱ 102985

⑲ 発 明 者 エリック・マーク・シ アメリカ合衆国ニューヨーク州エンディコット、ジェー
ユワツツ シン・レイスイ・ドライブ5エー番地

⑲ 発 明 者 ステマテイズ・ヴァシ アメリカ合衆国ニューヨーク州ヴェスタル、ヴェスタル・
リアデス ロード717番地

⑲ 出 願 人 インターナショナル・ アメリカ合衆国10504、ニューヨーク州アーモンク(番地
ビジネス・マシーン なし)
ズ・コーポレーション

⑲ 代 理 人 弁理士 頓宮 孝一 外1名

明 細 書

リーム複数データ(MIMD)パイプラインを使用する技術に係る。

1. 発明の名称 バイブライン式処理装置

2. 特許請求の範囲

入力命令を受取る受取り手段と、

それぞれが一意的な識別子により識別され、複数の入力命令を保持して実行する複数のパイプライン式処理手段と、

前記処理手段及び前記受取り手段に接続され、前記複数の入力命令を保持している処理手段の識別子及び該入力命令の情報を記録して、前記処理手段での命令実行を制御するテーブル手段と、

を具備するパイプライン式処理装置。

3. 発明の詳細な説明

A. 産業上の利用分野

本発明は計算機システムに係り、特に、単一命令ストリーム単一データ(SISD)アーキテクチャで動作するように設計されている、浮動小数点ユニットの如き機能ユニットで、複数命令スト

B. 従来技術とその課題

殆んどの計算機のプロセッサは何らかの形のパイプラインを利用している。パイプライン式プロセッサでは、命令ストリームの2以上の命令が同時に実行される。実行中の各命令はパイプの異なるステージにある。パイプライン式プロセッサは当然非パイプライン式プロセッサよりも高性能である。パイプライン方式には幾つかのタイプがある。その1つは単一命令ストリーム単一データ(SISD)パイプライン方式である。SISDタイプにおいては、個々の命令は単一のデータ・オペレーションとパイプライン結合される。しかし、SISDパイプライン方式を採ると多くのハザードに遭遇する。可能な最大の新データ・レートでパイプラインに入る時にハザードが生じる。ハザードは、構造ハザード及びデータ依存ハザードの2つに分けることができる。構造ハザードは、

2つのデータが同じハードウェアを使おうとした時、すなわちハードウェアの使用に関して衝突が起こった時に生じる。データ依存ハザードは、パイプラインの1つのステージで起こった事象がパイプラインの別のステージを介するデータ通過の可否を決定するときに生じ得る。例えば、パイプラインが2つのステージを有し、各ステージが単一メモリの使用を要求している時、一方のステージがメモリを使用中であれば、他方のステージはメモリが解放されるまで待つていなければならない。

別のタイプは、複数命令ストリーム複数データ(MIMD)パイプライン方式と呼ばれるものである。MIMDタイプでは、個々の命令ではなくて命令ストリームがパイプライン化される。MIMDタイプのパイプライン方式はハザードの問題を生じない。しかし、命令ストリームがパイプライン化されると云つても、或る命令ストリームの実行が完了しない限り、次の命令ストリームの実行を開始することはできない。従つて、MIMD

実行後、更新された命令はパイプライン回路14から記憶装置10に書き込まれる。命令ストリームの最後の命令が初期設定制御部12からパイプライン回路14へ送られ、その実行が終つて最後の更新された命令が記憶装置10へ送られると、別の命令ストリームがパイプライン回路14での実行のために、記憶装置10から初期設定制御部12へ読出される。

上述から明らかなように、第2図の構成においては、最初の命令ストリームのパイプライン化及び実行がパイプライン回路14で終わらない限り、次の命令ストリームを記憶装置10から初期設定制御部12へ読出すことはできない。これが従来のMIMDパイプライン方式の欠点である。

従つて本発明の目的は、上述の如き欠点のない新しいタイプのパイプライン(以下、動的MIMDパイプラインと云う)を計算機システムに導入することにある。

パイプライン方式の性能は、SISDパイプライン方式よりは上であるが、一時に1つの命令ストリームしか実行できないと云う原理によつて制限を受ける。

第2図を参照しながら、従来のMIMDパイプライン方式について説明する。

第2図において、記憶装置10は複数の命令ストリーム及び各命令ストリームの状態を記憶している。記憶装置10の出力には初期設定制御部12が接続され、そして初期設定制御部12の出力にはパイプライン回路14が接続されている。パイプライン回路14はハザード検出回路を持つていない。パイプライン回路14の出力は記憶装置10に接続されている。

動作時には、まず1つの命令ストリームが記憶装置10から初期設定制御部12へ送られる。初期設定制御部12は、受取つた命令ストリームの各命令を1つずつパイプライン回路14へ送る。これらの命令はパイプライン回路14の内部でパイプライン化され、一時に1つずつ実行される。

C. 課題を解決するための手段

本発明に従うパイプライン式処理装置は、入力命令を受取る受取り手段、入力命令を実行する複数のパイプライン式処理手段、及び処理手段での命令実行を制御するためのテーブル手段を含む。各処理手段(パイプ)は複数の命令を保持して実行することができ、それぞれ一意的な識別子(パイプ番号)により識別される。各処理手段に保持されている命令は様々な実行段階にあるため、処理手段における各命令の状況を正確に記録しておくことが必要である。そのため、テーブル手段(動的活動記録テーブル)は各処理手段に保持されている命令に関する情報を記憶する。この情報には、当該命令を実行する処理手段の識別子も含まれる。

上述の如きパイプライン式処理装置は、例えば浮動小数点ユニットに適用できる。一般に、浮動小数点ユニットはCPUの如き他の機能ユニットからの要求に応答して動作するが、命令実行で手

一杯の時は要求に応じられないことがある。また、何らかのハザードが存在していると、命令の実行を遅らせる必要がある。これらを監視するため、後述の実施例では、ハンドシェーク/大域ハザード回路が使用される。ユニットが使用中でなく且つハザードが存在しなければ、次の命令に対応する処理手段へ入れることができる。また、命令の長さや種類に応じてMIMDモード又はSISDモードを選択するMIMD/SISD切替え回路も設けられる。

D. 実施例

以下では、本発明に従う動的MIMDパイプラインを計算機システムの機能ユニットに組込んだ例を説明する。機能ユニットとしては浮動小数点ユニット(以下、FPUと略称)を取り上げるが、勿論本発明はこれに限定されるものではない。

計算機システムは、FPUの他に、キャッシュ、CPU、及びベクトル・プロセッサを含む。FPUは、キャッシュ、CPU又はベクトル・プロセ

ッサがその命令を実行すべき場合には、CPUは適当なPBUにプロセッサ・バス動作(PBO)信号を送る。例えば、CPUで解読した命令が長精度形式の浮動小数点乗算命令であれば、FPUで実行した方が適しているので、CPUはPBO信号をFPUへ送つて、この命令の実行を要求する。

FPUは2つの主要部、すなわちデータが実際に流れる部分と、命令が導入されて制御信号に変換される部分とを含んでいるが、本明細書で主に説明するのは後者の部分である。

本発明の動的MIMDパイプラインを利用するFPUの構成例を第1図に示す。

第1図において、FPU20のCバスは命令スタック21に接続される。命令スタック21の出力はデコーダ22に接続される。デコーダ22の出力はハンドシェーク/大域ハザード回路23、MIMD/SISD切替え回路24、及び初期設定回路25に接続される。回路23及び24の出力は初期設定回路25の入力に接続される。初期設定回路25の出力は動的活動記録テーブル27、

ッサからデータを直接受取る。命令はCPUから受取る。CPUは、キャッシュから来るデータの制御には関与しない。CPUは、(キャッシュから)データを要求する一方で、命令をFPUへ送る。データがキャッシュからアクセスされている間、CPUは命令をFPUへ送り続ける。その際、キャッシュからデータをアクセスするサイクルと、対応する命令をFPUへ送るサイクルとの間の同期は無視される。従つて、例えばサイクルNでFPUに到着したデータが、サイクルM(ただし $M \leq N$)でFPUに送られた命令に係るデータである場合がある。CPUは、Cバスと呼ばれるバスを介して、FPU及び他のユニット(例えばキャッシュ)に動作を要求する。Cバスは、CPUとFPUとの間で命令を転送する唯一の手段である。命令のOPコードの他に、ハンドシェーク制御信号もCバス上を転送される。CPUが要求を送る機能ユニットをプロセッサ・バス・ユニット(PBU)と呼ぶ。その1つがFPUである。CPUが自身で実行できない命令を検出し、PB

パイプライン機構26、及び複数の浮動小数点レジスタ(FPR)のアレイ28に接続される。ハンドシェーク/大域ハザード回路23の出力は例外処理回路19にも接続される。例外処理回路19の出力は動的活動記録テーブル27に接続される。パイプライン機構26も動的活動記録テーブル27及び例外処理回路19に接続され、またDバスに出力を発生する。Dバスは、データ・キャッシュ(図示せず)及びアーキテクチャ上で定義されたローカル・メモリであるFPRアレイ28に接続される。動的活動記録テーブル27の出力は、Dバス及びFPRアレイ28への出力ゲート動作を制御するのに用いられる。Cバスは、命令スタック21の他に、Dバス・スタック制御回路30にも入力を提供する。Dバス・スタック制御回路30の出力はDバス・スタック29に接続される。Dバス・スタック29はDバスから入力を受取る。Dバス・スタック29及びFPRアレイ28の出力は、データ・フローを開始するデータを発生する。

本発明の動的MIMDパイプラインは2つの経路、すなわち命令及び制御のための経路(Cバス経路)と、データ・フローのための経路(Dバス経路)に分けることができる。命令はCバスを介して受取られ、命令スタック21に置かれた後、デコーダ22で脱読される。データはDバスを介して第1図の動的MIMDパイプラインへ導入される。

ハンドシェイク/大域ハザード回路23はCPU(図示せず)へハンドシェイク信号を送り、大域ハザードを検出する。ハンドシェイク/大域ハザード回路23の詳細は第10図に示してあるが、これについてはあとで説明する。Cバスは、CPUと各PBU(FPU20を含む)との間で一組のハンドシェイク信号を転送する。FPU20がCバスを介して要求を受取った場合、その要求がCPUからのもので、且つFPU20が当該要求に関係する唯一のPBUであれば、ハンドシェイク/大域ハザード回路23は、肯定応答信号(ACK)、使用中信号又は割込み信号をCPUへ返

送する必要がある。ACKは、Cバスから要求を受取った時にFPUが使用中(ビジー)でなければ、FPUからCPUへ送られる。割込み信号は、データ例外が生じて状況ワードに重要情報が書き込まれていると、FPUからCPUへ送られる。使用中信号は、FPUが別の命令を受入れて実行することができない場合に、FPUからCPUへ送られる。

ハンドシェイク/大域ハザード回路23は、大域ハザードを検出すると、その存在を要する信号を初期設定回路25へ送る。また回路23は、(初期設定回路25と関連して)FPU20の応答を他のプロセッサ・バス・ユニット(PBU)へ送る。回路23は、命令ストリームの始め及び終りを検出するのにも役立つ。回路23は、実行中の他の命令に対する命令のデータ依存性によるハザード(データ・インターロック)の存在を検出する。

MIMD/SISD切替え回路24は、デコーダ22で脱読された入力命令に応じて、SISD

モード又はMIMDモードを設定する。入力命令が64ビットよりも長いオペランドを使用するものであるか、又は命令の実行が困難であれば、MIMD/SISD切替え回路24はSISDモードを選択し、さもなければMIMDモードを選択する。

実行が困難であると考えられ、SISDモードが選択される命令は次の通りである。

浮動小数点除算

固定小数点除算

平方根演算

拡張精度オペランドを用いる演算

SISDモードでの実行中は、当該命令の実行を除くと、どのような活動も行われない。これは、FPU20のハンドシェイク回路23からCPUへ送られる使用中信号を活動状態に保つことにより達成される。使用中信号が活動状態にあると、CPUは新しい要求をFPU20へ送れない。

以下に列挙する命令又はその如何なる組合せも、MIMD/SISD切替え回路24にMIMDモ

ードを選択させる。

浮動小数点演算

加算

比較

二分

ロード

乗算

記憶

減算

固定小数点演算—マイクロコード

乗算

その他の演算—マイクロコード

ロード

記憶

状況ワード

間接モード

再試行

次の命令は、MIMD/SISD切替え回路24にSISDモードを選択させる。

浮動小数点—マイクロコード

拡張精度加算

拡張精度乗算

除算

拡張精度除算

平方根演算

拡張精度丸めロード

固定小数点演算—マイクロコード

除算

初期設定回路25はパイプを起動し、動的活動記録テーブル27を更新する。初期設定回路25の詳細については、あとで第11図を参照しながら説明する。初期設定回路25はハンドシェーク／大域ハザード回路23と関連して、命令ストリームの始め及び終りを決定し、また何らかのデータ依存ハザードが存在するかどうかを決定する。命令解読後、デコーダ22からの出力により示される命令のタイプが、初期設定回路25及びハンドシェーク／大域ハザード回路23において、使用する適切なパイプの第1サイクル状況の完了状況（内部パイプ制御部26a～26dにより示さ

し、使用中信号での応答は命令ストリームの終りを示す。ハンドシェーク／大域ハザード回路23は、データ依存性によるハザードの存在を検出するのに用いられる。初期設定回路25は、動的活動記録テーブル27に新しいエントリを追加する。従つて、要約すると、初期設定は、ハンドシェーキング、活動記録ファイルの更新、及びもしデータ依存ハザードが生じていればその処理、から成る。

第1図の動的MIMDパイプラインは4つのパイプライン回路（パイプ1～パイプ4）26a～26dを含む。従つて命令のカテゴリも4つある（1つのパイプ当り1つのカテゴリ）。

Dバス上のデータはFPRアレイ28又はDバス・スタック29により処理される。Dバス・スタック29はDバス・スタック制御回路30の制御を受ける。

例外処理回路19は、例外が生じているかどうかを調べる。命令実行中に生じ得るデータ例外のタイプは次の通りである。

れる）と比較される。大域ハザードがないことが動的活動記録テーブル27により示され、当面の内部ハザードがないことがハンドシェーク／大域ハザード回路23の出力により示されると、命令の初期設定が行われる。ハンドシェーク／大域ハザード回路23が使用中信号を発生していると、初期設定回路25は如何なる初期設定も行わない。

初期設定は、適切なパイプの状況制御の開始、及び動的活動記録テーブル27への新しいエントリの入力を含む。初期設定の状況をCPUに知らせるのはハンドシェーク／大域ハザード回路23である。回路23は、ACKをCPUへ送ることにより、命令の処理が開始されたことを示し、また使用中信号をCPUへ送ることにより、FPU20が命令を受取つたが、それ程多くの命令を処理できないため、入力命令のパイプを停止することを示す。前述のように、回路23及び25は命令ストリームの始め及び終りを決定する。まだ命令ストリームにない命令に対するACK及び使用可（使用中でない）は命令ストリームの始めを示

指数オーバーフロー例外

指数アンダーフロー例外

浮動小数点除算例外

固定小数点除算例外

有効数字例外

平方根演算例外

これらの例外のうちの1つを起こす命令が検出されると、この命令の後で受取られたすべての命令は、例えそれらが既に実行中であつたとしても、あたかも受取られなかつたかの如き効果を与えるため、取消さなければならない。これは、動的MIMDアーキテクチャが維持しなければならないSISDアーキテクチャの性質である。命令の取消しは、例外を起こした命令の完了後に動的活動記録テーブル27の中のすべての有効ビットをゼロに変えることにより為される。更に、CPU及び他のユニットは割り込みを知らされ、CPUが割り込み処理ルーチンを開始するまでそれらの命令を取消さなければならない。

計算機システムのFPUに配置される第1図の

動的MIMDパイプラインはCバスから命令を受取り、FPUは他のPBUと同様に、「ACK」、「使用中」及び「割込み」の如き特定のハンドシェーク信号を送ることによつて応答する。CPUはパイプライン・モードで働き、サイクル毎にPBOコマンドを、ACKが返されたかどうかには無関係に送出するので、PBUは次のPBOの実行に進む前に、最後のPBOに対するACKが返されたかどうかを調べなければならない。PBUはスマート・インタフェースを含んでおり、それを用いて、他のPBUとCPUとの間のハンドシェーク状態をチェックする。PBUは、PBOを受取った後のサイクルで、3つのハンドシェーク信号のうちの1つを回路23からCPUに送る必要がある。PBUは、ハザードに遭遇すると、使用中信号をCPUに送る。その時PBUは、受取った命令及びその次の命令を命令スタック21に保持しており、かくしてCPUからの命令の順序が維持される。

第3図の(a)に示すように、命令スタック21

のビットを示し、(c)はマイクロコード・モード時におけるCバス上のビットを示している。

前述のように、命令スタック21はCバス・スタック21a及びその出力に接続されたCバス・レジスタ21bを含む。Cバスと同じく、命令スタック21は多くとも2つの命令につき25ビットの情報を保持する。各ビットの意味は次の通りである。

ビット0—FPU20が配線モードにあるかマイクロコード・モードにあるかを示すPBOビット。配線モード(ビット0=0)においては、例外はCPUに報告される。マイクロコード・モード(ビット0=1)においては、例外は状況ワード(第8図参照)に記憶されるが、報告されることはない。

ビット1(FP)—当該命令がFPU20で実行されねばならないことをFPU20に知らせるFPU要求ビット。

ビット2(IPU)—命令の解釈をキャッシュに知らせるIPU/キャッシュ要求ビット。

はCバス・スタック21a及びCバス・レジスタ21bを含む。受取った命令はCバス・レジスタ21bに保持され、その次の命令はCバス・スタック21aに保持される。使用中信号を発生させるハザードに遭遇しない限り、命令はスタックされない。FPU20は、処理能力の範囲内であれば、どれ程多くの命令でも受入れるが、CPU程多くの情報を含まない。と云うのは、CPUはその命令バッファの内容から、何らかの問題が生じそうだと判断すると、命令をバス・ユニットへ送る前にその命令を停止させることができるからである。FPU20及び他のバス・ユニット(例えばデータ・キャッシュ)での実行を要求するPBOがCPUから送られる場合、FPU20はデータ・キャッシュによる命令実行の開始を阻止できない。従つて、FPU20によるパイプライン化の最も効率的な方法は、ハザードに遭遇するまでにできるだけ多くの処理を行うことである。

第3図において、(a)は命令スタック21の構成を示し、(b)は配線モード時におけるCバス上

ビット3(VP)—ベクトル・プロセッサ要求ビット。

ビット4~10—命令のOPコード。

ビット11~13(F1)—オペランド1の符号化されたFPRアドレス。

ビット14~16(F2)—オペランド2の符号化されたFPRアドレス。

ビット17~19(TAG又はSRC)—配線モードにおいては、これらのビットは、例外発生に伴つて状況ワードに記憶される割込みタグ(TAG)を表わし、マイクロコード・モードにおいては、ソースPBU(SRC)を識別する。割込みタグは、CPUの命令スタックにある命令を一意に識別する。

ビット20~22(DST)—マイクロコード・モードにおいて宛先PBUを識別する。

ビット24(P)—命令の妥当性を検査するためのパリティ・ビット。

Cバス上の命令は上述の25ビットの形で命令スタック21へ導入される。なお、第3図の(b)

及び(c)の中の斜線部分は予約フィールドを表わしている。

動的活動記録テーブル27の構成を第4図に示す。第4図の例では、動的活動記録テーブル27は、最大8個の命令までそれぞれ17ビットの情報を記憶する。これは、入力命令が4つのパイプライン回路26a~26dのうちの1つに入って完了する必要があるときに使用される。命令はスタックされるので、テーブル27は1以上の命令ストリームの命令実行の完了を順序づける手段を提供する。Cバスは一時に1つの命令しか送れないので、この結果として命令の開始時間が決まる。1以上の命令ストリームの命令の実行は完了までに複数のサイクルを要することがあり、またパイプも複数存在しているので、複数の命令を同時に実行することが可能である。アーキテクチャ上の制約から、割込みが起こった時に命令がシーケンシャルでなかつたなら、結果が予測できないことがあるので、命令完了の順序は維持されねばならない。従つて、順序づけ情報及び完了情報をテ-

とを示す。

ビット11~13 (INT TAG) - CPUのスタックにおける命令を一意的に識別する割込みタグ。

ビット14~16 (PSW PTR) - 命令の再試行ポインタを識別する。

複数のパイプライン回路26a~26dを順序づけるという点でパイプ番号 (PIPE NO) は重要である。パイプが1つだけであれば、順序づけは殆んど問題にならないが、複数パイプのシステムでは、追跡情報を維持しておかねばならない。書き込みアドレス (WR ADDR)、書き込みタイプ (WT) 及び結果長 (LEN) は命令を完了させるのに有用である。割込みタグ (INT TAG) は、例外が生じた時に記憶され、この例外を起こした命令を識別する。もしそれがSISD命令であれば、パイプの終りに有効データがあるかどうかを見る代りに、サイクルを計数するカウンタによつて命令の完了が感知される。最も重要なビットは、対応する命令が有効かどうかを示

す。ブル27に保持しておく必要がある。テーブル27は次のような情報を記憶する。

ビット0 (V) - 有効ビット。

ビット1~3 (WR ADDR) - 書き込みアドレス。

ビット4~5 (PIPE NO) - パイプ番号 (00=加算、01=乗算、10=ロードRX、11=その他)。

ビット6 (H) - 1であれば配線FPU要求であることを示し、0であればマイクロコード命令であることを示す。

ビット7 (WT) - 書き込みタイプの命令かどうかを示す。

ビット8 (M/S) - 1であればMIMD命令タイプであることを示し、0であればSISD命令タイプであることを示す。

ビット9 (EXT) - 拡張精度結果を書込むかどうかを示す。

ビット10 (LEN) - 1であれば結果が長精度であることを示し、0であれば短精度であるこ

と有効ビット (V) である。有効ビット (V) は例外が生じるとクリアされる。有効ビット (V) は命令完了時にもクリアされるが、その場合はスタックが上方にシフトされる。従つて、動的活動記録テーブル27において有効ビット (V) をクリアすることにより、FPU20中のすべての未了命令を迅速に取消することができる。

パイプライン機構26の構成を第5~8図に示す。第5図は、加算、減算、除算、比較、平方根演算等の加算タイプの命令に用いられる加算パイプを含むパイプライン回路26aの構成を示している。第5a図の加算パイプは3サイクルでその機能を遂行する。第6a図は乗算命令に用いられる乗算パイプを含むパイプライン回路26bの構成を示しており、これは5サイクルでその機能を遂行する。第7図はRXタイプのロード命令に用いられるパイプライン回路26cの構成を示しており、これは2サイクルでその機能を遂行する。第8図は他のすべての機能 (普通は補助レジスタや状況レジスタの書き込み又は読取り) を遂行する

ためのパイプライン回路26dの構成を示している。

パイプライン回路26a~26dはそれぞれ制御部及びパイプ部(1~4)を含んでいる(第1図参照)。制御部は、関連するパイプ中の流れをできるだけ速くまで行かせ且つFPRアレイ28がインターロックされる時を感知することにより、及びどこに良好なデータがあるかを調べることに、関連するパイプの内部を制御する。MIMDモードでは、パイプ1~4が異なつた長さを持つているため、これらのパイプの全体的な制御が複雑になる。

パイプ内部には関連する状況フィールドを有する幾つかのレジスタがある。第5a図の加算パイプの場合、第5b図に示した状況フィールドの各ビットの意味は次の通りである。

ビット0~2(ADDR) - オペランドのFPRアドレス。

ビット3(VI) - パイプ中の当該ステージにおける命令が有効かどうかを示す。

のビットS(VR)は、パイプの当該ステージが有効な命令を持つていなくてもデータ(結果)が有効であるかどうかを示す。他のパイプ3及び4は短いので、状況情報は不要である。

パイプの各ステージに関する状況情報は、次のステージにその有効性を知らせるものであり、もし問題がなければ、次のステージは次のサイクルで有効になる。このように、状況情報は、問題の有無を判断したり、命令のデータをパイプ中のできるだけ速くまで流すようにしたりするのに有用である。当該パイプによるDバスへのデータ出力及び実行後に、当該パイプは、そのパイプ番号が動的活動記録テーブル27の最も古いエントリ中のパイプ番号(PIPE NO)と一致するのを待つ。一致した時、当該パイプは完了を許され、かくして命令完了の同期が維持される。

第5a図に示した加算パイプ(パイプ1)は、整列レジスタ34、FAレジスタ31、FBレジスタ32、Aレジスタ35、Bレジスタ36、加算器37、FSレジスタ33、Sレジスタ38、

ビット4(VD) - 関連するレジスタ中のデータが有効かどうかを示す。

ビット5(M/S) - MIMDモードかSISDモードかを示す。

ビット6(RX) - 命令がRXタイプかRRタイプかを示す。

ビット7(2BY) - これが1になつていると、2サイクルのパイパスが行われていることを示す。

第5a図の乗算パイプの場合は、更に次のような4ビットの状況情報を含む。

ビット8(EXT) - 拡張精度結果かどうかを示す。

ビット9(LI) - オペランドが長いかどうかを示す。

ビット10(FLP) - 1であれば浮動小数点乗算であることを示し、0であれば固定小数点乗算であることを示す。

ビット11(INTL) - Yのオペランドがインターロックされるかどうかを示す。

なお、乗算パイプにおいては、状況フィールド

及び事後正規化レジスタ39を含んでいる。第5b図に示した状況フィールドは、FAレジスタ31、FBレジスタ32及びFSレジスタ33に関連している。

第5a図の加算パイプは3サイクルでその機能を遂行する。第1サイクルでは、データがFPRアレイ28又はDバスから検索され、整列レジスタ34で整列操作が行われ、そしてオペランドがAレジスタ35及びBレジスタ36にラッチされる。第2サイクルでは、加算器37で実際の加算が行われ、その結果がSレジスタ38に書き込まれる。最後の第3サイクルでは、事後正規化レジスタ39が必要に応じて先行ゼロをシフトアウトし、データをFPRアレイ28に戻す。

上述の機能は、第5a図のパイプが加算命令を処理するときのものであるが、加算タイプに属する他の命令の場合は、内部バイパス制御のために他のレジスタも使用される。複数の異なつた命令を処理し得る第5a図のパイプを維持し制御するためには、3つの主制御レジスタが必要である。

それらは、F Aレジスタ31、F Bレジスタ32及びF Sレジスタ33である。

前述のように、レジスタ31、32及び33の状況フィールドは以下のビットを含んでいる。

1. インターロックされ得るオペランドを見つけるのに用いられるF P Rアドレス・ビット(ADDR)。
2. パイプ中の当該ステージが命令に対して有効であることを示すのに用いられる有効命令ビット(VI)。
3. 関連するデータ・レジスタが有効であることを示す有効データ・ビット(VD)。
4. 命令の終りを知らせるMIMD/SISDパイプ標識(M/S)。MIMDモードの場合は、最終ステージが有効で、完了について観合がなければ、命令の終りである。SISDモードの場合は、命令がパイプ中を何回かループすることがあるので、もう少し複雑になる。
5. 命令がR Xタイプであることを示すビット(RX)。これは、F Bレジスタ32に関して

128からロードするのであれば、オペランド2はF P Rアレイ28から脱出されて一時レジスタに記憶される。

サイクル2ーオペランド2が一時レジスタ又はDバスからYレジスタ49にロードされ、同時にオペランド1の3倍乗算が3Xハードウェア47で行われ、結果がXB/3Xレジスタ48の3X部に書込まれる。更に、XAレジスタ46の内容が直接XB/3Xレジスタ48のXB部にロードされる。

サイクル3及びサイクル4ーこれらは乗算器の実際の実行サイクルで、M1サイクル及びM2サイクルと呼ばれ、M1ハードウェア50及びM2ハードウェア51を使用する。M1サイクルとM2サイクルの間にレジスタが介在することではなく、従ってXB/3Xレジスタ48及びYレジスタ49は、データがPレジスタ52にラッチされるまで、これら2サイクルの間保持されねばならない。

サイクル5ーPレジスタ52からF P Rアレイ

は、そのアドレス・ビットが無効で、入力データ(まだ有効でなければ)のためにデータ・バスを監視する必要があることを示す。

6. 2サイクル・バイパスの第1サイクルを示すビット(2BY)。インターロックされたデータが見つかった場合、それを取り出すのに2サイクルを要することがある。

第6a図の乗算パイプ(パイプ2)は、F X Aレジスタ41、F Y Sレジスタ42、F X Bレジスタ43、F Yレジスタ44、F Pレジスタ45、X Aレジスタ46、3 Xハードウェア47、X B/3 Xレジスタ48、Yレジスタ49、M1ハードウェア50、M2ハードウェア51、及びPレジスタ52を含んでいる。この乗算パイプは、もしハザードに遭遇しなければ、5サイクルでその機能を遂行する。

サイクル1ーオペランド1がF P Rアレイ28からXAレジスタ46にロードされる。バス幅の制限から一時的に1つのオペランドしかロードできないので、もしオペランド2もF P Rアレイ

28への書込みが行われる。本実施例では、チップ間に1本の8バイト・データ・バスしか設けていないので、拡張精度結果の場合には、第2の書込みサイクルであるサイクル6が後に続く。

乗算パイプを制御する制御レジスタは、XAレジスタ46の状況維持するF X Aレジスタ41、R Rタイプの命令の場合にオペランド2を最初に受取る一時レジスタの状況維持するF Y Sレジスタ42、X B/3 Xレジスタ48の状況維持するF X Bレジスタ43、Yレジスタ49の状況維持するF Yレジスタ44、及びPレジスタ52の状況維持するF Pレジスタ45である。これらの制御レジスタないし状況レジスタは、第6b図に示す12ビットの情報を維持する。これらのビットは次の通りである。

1. インターロックされ得るオペランドを見つけるのに用いられるF P Rアドレス・ビット(ADDR)。
2. パイプ中の当該ステージが命令に対して有

効であることを示すのに用いられる有効命令ビット(VI)。

3. 関連するデータ・レジスタが有効であることを示す有効データ・ビット(VD)。

4. 別のチップ上でローカル作業用記憶域を作り出すのに用いられる有効結果ビット(VR)。RR命令の場合、オペランド2は、別の命令がその内容を変更するまでは、乗算後もYレジスタ49で有効になつている。有効結果ビットは、Yレジスタ49にあるデータが定義されたアドレスに関して有効であることを示す。また、RR命令及びRX命令の場合、乗算の結果を含むPレジスタ52は、別の乗算がこのパイプで行われるか、又は別の命令がオペランド1のFPRアドレスを変更するまでは、オペランド1によつてアドレス指定されるFPRと同じである。これは性能を上げる上で極めて重要である。ロードを少しでも減らすことができれば、性能の向上につながる。

5. 命令がRXタイプで、そのアドレス・ビッ

トが無効であることを示すビット(RX)。入力データがまだ有効でなければ、そのためにデータ・バスを監視する必要がある。

6. 2サイクル・バイパスの第1サイクルを示すビット(2BY)。インターロックされたデータが見つかった場合、それを取り出すのに2サイクルを要することがある。

7. 拡張精度結果をFPRアレイ28に書戻さなければならないことを示すビット(EXT)。

8. 長いオペランドがレジスタにあることを示すビット(LI)。

9. バイプ中の当該ステージにある命令が浮動小数点命令であることを示すビット(FLP)。

10. Yレジスタ49のオペランドに対するインターロックを示すビット(INTL)。

FXAレジスタ41及びFYSレジスタ42は、サイクル1で初期設定回路25によりセットされる。XB/3Xレジスタ48に関して競合がなければ、FXBレジスタ43はFXAレジスタ41からセットされる。Yレジスタ49に関して競合

がなければ、FYレジスタ44は初期設定回路25又はFYSレジスタ42によりセットされる。競合は次のような形をとり得る。

M1有効、又は

M2有効且つPレジスタ競合、又は

XAが前の乗算に対し既に有効で且つXB競合
FPRレジスタ45は、M2が有効で且つPレジスタ52についての競合がないときに、FXBレジスタ43によりセットされる。FPRレジスタ45の有効結果ビット(VR)は、FPRアレイ28への他の書込みに依存しているので、別に維持される。かくして、内部ハザードに遭遇するまで、又は乗算パイプ中の命令の完了を妨げる外部ハザードに遭遇することなくPレジスタ52が有効になるまで、パイプ中をできるだけ遠くまでデータを流す内部パイプ制御により乗算パイプが維持される。

RXタイプのロード命令に用いられるパイプ3の構成を第7図に示す。パイプ3は2サイクルしか要せず、その間データは単にパイプ中を流れる

だけであつて、その他の処理は行われない。命令はデコーダ22で解読され、FPUはDバス・スタック制御回路30からのDバス有効信号を待つ。サイクル1では、データがDバス・スタック29のデータ・レジスタ(DREG)91に受取られる。このデータは、サイクル2の間にFPRアレイ28へ送られる。これを監視する制御部は、DREG有効レジスタ71を含むDバス・スタック制御回路30である。

上述以外の機能(雑機能)を遂行するパイプ4の構成を第8図に示す。このパイプは、Dバス・スタック29の一部を成すDREG91の出力に接続された間接アドレス・レジスタ61、同じくDREG91の出力に接続された状況ワード・レジスタ62、Cバス・レジスタに接続された再試行状況レジスタ63、及びCバス・レジスタに接続された間接アドレス・モード・レジスタ64を含む。状況ワード・レジスタ62及び再試行状況レジスタ63の出力はDREG91及びFPRアレイ28に接続される。データは、Dバスから間

接アドレス・レジスタ61及び状況ワード・レジスタ62にロードされる。間接アドレス・レジスタ61はマイクロコード・モードで使用され、FPRアドレスを含む。状況ワード・レジスタ62はFPUの状況、例えば例外や検査機構の状況を維持する。

第8図のパイプ4で遂行される雑機能の命令は5つの基本グループに分けられる。

1. 読取り(簡単なビット操作を伴うことがある)に続いてFPRアレイ28への書き込みを基本的に1サイクルで行うロードRRグループ。
2. FPRアレイ28以外の間接アドレス・レジスタ61や状況ワード・レジスタ62へのロードを行うロードRXグループ。
3. この第3グループの命令は2サイクルで実行される。サイクル1では、DREG91をロードする一方、DREG有効レジスタ71が有効になるのを待ち、サイクル2では、DREG91からDREG有効レジスタ71へのロードを行う。Cバスからの情報でレジスタをセット

する命令もこのグループに含まれる。例えば、再試行情報を維持する再試行状況レジスタ63、及びアドレス指定のためのモード・ビットを維持する間接アドレス・モード・レジスタ64は、このような命令によりセットされる。

4. サイクル1でFPRアレイ28の読取りを行ってDREG91に書き込み、サイクル2でデータをDバスへ出力してDバス有効を知らせる記憶RXグループ。

5. 最後の第5グループは、FPRアレイ28以外のレジスタの内容を2サイクルでDバスに置く記憶タイプの命令を含む。第5グループの命令で読取られる2つのレジスタは、状況ワード・レジスタ62及び再試行状況レジスタ63である。

Dバス・スタック制御回路30のレジスタを除くと、パイプ4の動作に関連する制御レジスタはない。

Dバス・スタック29及びDバス・スタック制御回路30の構成を第9a図及び第9b図に示す。

Dバス・スタック制御回路30は、DREG有効レジスタ71、DCバス・レジスタ71、デコーダ73、DS5レジスタ74、DS4レジスタ75、DS3レジスタ76、DS2レジスタ77、DS1レジスタ78、S2有効レジスタ79、S1有効レジスタ80、及びMUX選択回路81を含む。Dバス・スタック29は、Dバスに接続されたS2レジスタ93、S2レジスタ93の出力及びDバスに接続されたS1レジスタ92、並びにS1レジスタ92の出力及びDバスに接続されたDREGレジスタ91を含む。

DS1からDS5までのレジスタ78~74は次のようなビットを含んでいる(第9b図参照)。

ビット0~2(IADDR)ー命令アドレス。3ビットの命令アドレスがゼロでなければ、境界間ロードが生じることを示す。境界間ロードは2つのDバス有効信号(データの各部につき1つ)を必要とする。

ビット3及び4(PIPE NO)ーデータを送るべきパイプを一意的に識別するパイプ番号。

ビット5(VI)ー有効命令ビット。

ビット6(DR)ー当該命令のためのデータがDバス・スタック29のDREG91にあることを示す。

ビット7(S1)ー当該命令のためのデータがDバス・スタック29のS1レジスタ92にあることを示す。

ビット8(S2)ー当該命令のためのデータがDバス・スタック29のS2レジスタ93にあることを示す。

ビット9(EXE)ー命令が実行中で、適切なパイプが初期設定されたことを示す。

データを受取った時、そのデータを前述のようにしてパイプ中を流す適切な命令はどれかを見出さなければならない。これは混沌状態を引き起こす。次に、この混沌状態について詳述する。

通常のRRタイプのオペレーションにおいては、インターロックが存在しない限り、データはFPRアレイ28から来る。その場合、ローカル・パイプ制御部は、すべてのパイプの状況レジスタの

アドレス・フィールドを比較することにより適切なデータを見つける。RXタイプのオペレーションの場合はもう少し難しい。オペランド1は同じようにして見つけれられるが、オペランド2はデータ・バスから来る。データ・キャッシュを含む他のバス・ユニットは、FPUが命令を受取るのと同時にデータを供給するよう要求され、その時そのバス・ユニットが使用中であれば、FPUによる命令及び関連データの受取りが同期的に行われず、その結果として、データのアンダーフロー又はオーバーフローが生じ得る。Dバス・スタック制御回路30は、DREG91が適切なパイプに対して有効になると、その旨をローカル・パイプライン機構26に知らせる。これは、データを分離する際に、特に幾つかのパイプでデータが不足している状態でデータ・バスが有効になった時、極めて重要である。従つて、Dバス・スタック制御回路30はデータと命令を順序づける。これが終ると、パイプライン機構26はパイプを通してデータを移動させ、動的活動記録テーブル27が

れた時に生じる。メモリは読出したデータを直ちに適切なパイプへ入れることはできないので、データをFPUへ送る。従つて、データをスタックすること、及び最終的にデータを送るべきパイプのパイプ番号でデータを識別することが必要である。

ハンドシェーク/大域ハザード回路23の構成を第10図に示す。この回路23は、パイプ1新命令競合回路101、パイプ2新命令競合回路102、パイプ3新命令競合回路103、パイプ4新命令競合回路104、大域競合回路105、ハンドシェーク組合せ論理回路106、及び割込みハンドシェーク組合せ論理回路107を含む。回路101～104はすべてパイプライン機構26及びデコーダ22(第1図参照)に接続される。大域競合回路105は動的活動記録テーブル27に接続され、他のユニットからのハンドシェーク信号に応答して、FPU使用中信号をCPUへ送つたり、ユニット使用中信号を初期設定回路25へ送つたりする。ハンドシェーク組合せ論理回路

命令の完了を認めるのを待つ(あとで述べるように、命令の完了は、動的活動記録テーブル27において最も古い命令、すなわち一番下のエントリのパイプ番号と当該パイプのパイプ番号とを比較することにより検出できる)。このようにして、データの流れが始めから終りまで制御される。更に、メモリはデータが何時要求されたかには無関係に、可能になつた時点でデータを供給するため、データのオーバーフロー又はアンダーフローが生じる可能性があり、この点でもDバス・スタック制御回路30が必要である。アンダーフローは、幾つかのパイプが初期設定されているにもかかわらずメモリからのデータ供給が遅い時に生じる(メモリからのオペランドを必要としないRXタイプの場合を除く)。複数のパイプがデータを待っている状態でデータが到着した時には、そのデータをどのパイプに送るかを正しく選択する必要がある。オーバーフローは、1以上のパイプ一杯であつて、メモリからのデータを使用する複数の命令がCPUからFPU及びメモリへ同時に送ら

106は大域競合回路105に接続され、他のユニットからのハンドシェーク信号に応答して、FPU ACK信号をCPUへ送つたり、有効命令信号を初期設定回路25へ送つたりする。割込みハンドシェーク組合せ論理回路107はハンドシェーク組合せ論理回路106及び例外処理回路19に接続され、FPU割込み信号をCPUへ送る。

第10図の回路23へ入力を供給するのは、他のユニット(PBU)、パイプライン機構26、動的活動記録テーブル27、及びデコーダ22である。デコーダ22からの入力は、新しい命令がCバス上にあるかどうか、及びどのパイプで新しい命令を実行しなければならないかを回路23に知らせる。パイプライン機構26からの入力は、各パイプについてその第1ステージで何らかのハザードが生じているかどうかを示す。他のユニットからの入力(ハンドシェーク信号)は、FPUの外部にハザードが存在するかどうかをみるために回路23により監視される。動的活動記録テーブル27からの入力は、何らかの内部大域ハザ-

ドが生じているかどうかを示す。ハザードがなく、新しい命令を実行すべきパイプで競合がなければ、FPU ACK信号がハンドシェーク組合せ論理回路106からCPU及び他のユニットへ送られる(命令が2以上のバス・ユニットに対するものであつて、ACKが抑止される場合を除く)。新しい命令があつても、ハザード又は競合が存在していると、ユニット使用中信号が初期設定回路25へ送られ、FPU使用中信号がCPU及び他のPBUへ送られる。更に、新しい命令は命令スタック21のCバス・レジスタ21bに保持される。この命令は、ACKを出せるサイクルが生じるまで、後続のサイクルの間新しい命令とみなされる。初期設定回路25へ送られる有効命令信号は、Cバス・レジスタ21bにある命令が適正CPUハンドシェークを交してFPUで実行すべきであることを示す早期信号である。これはハザード及び競合には依存しない。回路23から送られる最後の信号は、例外条件を示すFPU割込み信号であり、CPUへ送られる。例外条件は前の命令

ータの長さに関する長さ情報(短いデータはゼロ充填が必要である); FPRアレイ28への書込みを行う時にアドレス情報を供給する書込みタイプ情報; FPRアレイ28からデータを取り出す時のアドレス情報に関するFPRアドレス情報; 初期設定回路25へ入力される命令がRXタイプかどうかを示すRX命令情報(RXタイプの命令では、命令のデータの2番目の部分は、FPRアレイ28ではなくて主記憶装置から供給される); パイプ1~4の制御に必要なその他の情報; 及びパイプ1~4のうちの1つを識別するパイプ番号。デコーダ22がパイプ番号を供給する理由は、各パイプが特定の命令に関連して機能するように特殊化されており、どのタイプの命令が初期設定回路25へ入力されるか、従つてどのパイプに入力命令を入れるべきかをデコーダ22が知っているからである。これは、命令解釈というデコーダ22の機能から考えて当然である。MIMD/SISD切替え回路24の出力は、MIMD/SISD切替え回路24の切替え位置に関する情報を

に対する他のPBUの応答に依存するので、FPU割込み信号は例外条件を調べる例外処理回路19からゲートされねばならない。パイプライン式の計算機においては、複数の命令が出されて同時に実行されるため、同じサイクルで複数の例外が生じる可能性がある。命令セットはSISDであるから、命令が順次に行われているかの如くに例外を処理しなければならない。これは、ハンドシェーク制御から決定される条件でFPU割込み信号をゲートすることにより為される。このようにして、ハンドシェーク/大域ハザード回路23によるハンドシェーキング及び大域ハザード検出が達成される。

初期設定回路25の構成を第11図に示す。初期設定回路25は、ハンドシェーク/大域ハザード回路23、デコーダ22、及びMIMD/SISD切替え回路24からの出力にตอบสนองする。ハンドシェーク/大域ハザード回路23からの出力は有効命令信号及びACK信号を含む。デコーダ22からの出力は次のような情報を含む: 命令のデ

与える。

初期設定回路25は、それぞれ同じ選択信号及び同じ情報信号を受取る5個のマルチプレクサ(MUX)111~115を含む。各マルチプレクサは、選択信号によつて選択されると、情報信号を通過させる。選択信号は、(1)ハンドシェーク/大域ハザード回路23からの有効命令/ACK信号出力、及び(2)デコーダ22からのパイプ番号出力により構成される。情報信号は、(1)長さ、(2)書込みタイプ、(3)FPRアドレス、(4)RX、(5)MIMD/SISD、及び(6)その他の情報を含む。マルチプレクサ111~115のうちの1つがパイプ番号信号及び有効命令/ACK信号を含む選択信号により選択されると、長さ情報、書込みタイプ情報、FPRアドレス情報、RX命令情報、MIMD/SISD切替え情報、及びその他の情報を含む情報信号がパイプライン回路26a~26dのうちの1つ又は動的活動記録テーブル27へ送られる。

次に、第1図~第8図を参照しながら、本発明

に従う動的MIMDパイプラインの動作について述べる。

第1図において、それぞれ複数の命令を含む複数の命令ストリームが、パイプライン機構26での実行を待っているものとする。これらの命令ストリームの中から第1図の動的MIMDパイプライン(FPU)20へ入力される複数の命令が選択回路(図示せず)により選択され、1つずつCバスを介して動的MIMDパイプライン20へ入力される。命令は命令スタック21に受取られ、デコーダ22で解説される。デコーダ22は、受取った命令がFPUで実行できるものであるかどうかを調べる。命令の受取りはハンドシェーク/大域ハザード回路23により確認され(ACK信号がCPUへ送られる)、命令はできるだけ早い機会に実行される。CPUは、FPUの連続処理が可能であることを前提にして、命令を連続的に第1図のFPUへ送る。

データ及び1以上の命令がFPUへ入力できる状態にあるものとする、FPUは、デコーダ2

かを判断する。前述のように、各パイプは特定カテゴリの命令を処理するように特殊化されており、従って新しい入力命令がデコーダで解説されて、命令のタイプが決定されると、使用する特定のパイプを識別できる。初期設定回路25はデコーダ22からパイプ番号を受取り、マルチプレクサ111~115を介して対応するパイプへ新しい命令を送る。命令がパイプ1~4のうちの1つに入ると、そのパイプの識別情報を記録すべく動的活動記録テーブル27が更新される。パイプ1~4はフィードバック形式で動的活動記録テーブル27に接続されているので、パイプ中での特定の命令の実行状況は継続的に動的活動記録テーブル27に記録される。従って、デコーダ22及び初期設定回路25で、別の新しい命令が特定のパイプに対応していると識別され、そのパイプに入る準備ができると、そのパイプに内部ハザード(例えば、データ・インターロック又はパイプ満杯)が存在しているかどうかを調べるため、該パイプの最初のステージに関連するパイプ制御部に記憶さ

2で命令を1つずつ解説し、パイプラインがSISDモードで動作するのかMIMDモードで動作するのかをMIMD/SISD切替え回路24で判断し、何らかのハザードが存在しているかどうかをハンドシェーク/大域ハザード回路23で調べ、どのパイプが命令を実行するのかをデコーダ22及び初期設定回路25で決定し、適切なパイプが何時完了してその中に入力命令を入れられるかを動的活動記録テーブル27で判断し、これらのことがすべて終ると、FPUは命令を最大速度で一時に1つずつパイプライン機構26へ入れることによつて命令を実行しようとする。初期設定回路25及びハンドシェーク/大域ハザード回路23は、命令ストリームが始まろうとしているのか終わろうとしているのかを判断し、命令ストリーム及び単一命令の正常終了又は異常終了をCPUに知らせる。ハザード存在の可能性があるため、初期設定回路25及びハンドシェーク/大域ハザード回路23は、パイプ1~4のうちの1つがいつ利用されるのか、及び命令がいつ必要になるの

れている情報が読取られる。内部ハザードが存在していると、新しい命令はこのパイプに入れない。その時、この特定のパイプのパイプ番号が動的活動記録テーブル27の一番下のエントリのパイプ番号欄(PIPE NO)にあるかどうか調べられる。例えば、特定のパイプが番号Xにより識別され、3つのパイプライン・ステージを有していて、動的活動記録テーブル27がパイプ番号欄にパイプ番号Xを含む3つのエントリを有していると、パイプXは命令で一杯であり、従って新しい命令を入れるためには、パイプX中の最も古い命令を完了させなければならない。テーブル27における最も古い(一番下の)エントリのパイプ番号欄がXになつてい、パイプX中の最も古い命令を完了させることができ、これにより、新しい命令をパイプXに入れられるようになる。要約すると、テーブル27の働きは、パイプ1~4の制御部と一緒になつて、すべてのパイプが適正に利用され且つ命令が始めから終りまで適正に実行されるように、継続的な補助を与えることにあ

る。各パイプは特定カテゴリの命令を実行し、その外部的な制御は動的活動記録テーブル27及び初期設定回路25が受持ち、内部的な制御は内部パイプ制御部が受持つ。各パイプは、複数の命令ストリームに属する複数の命令を実行することもできる。

次に、第1図及び第4図を参照しながら、動的活動記録テーブル27の動作について説明する。

ACK信号がハンドシェイク/大域ハザード回路23から送られるか、又は使用中信号が回路23によつて落とされると、入力命令に関する情報が動的活動記録テーブル27の1つのエントリに入れられる。第4図の例では、テーブル27は8個のエントリ(1~8)を持っている。このテーブル27は、入力命令をどのように完了させるかを決定するのに必要な幾つかのパラメータを含む。キー・パラメータは、当該命令を実行するパイプを示すパイプ番号である。アーキテクチャ上の制約から順次的な実行に見せかける必要があるため、動的活動記録テーブル27はサイクル毎に脱出さ

れ、入力命令の実行を次に完了するのはどのパイプかを示す。各サイクルでテーブル27から読出された情報は、選択されたパイプが完了を待っているかどうかを見るために、そのパイプの内部状況(状況レジスタにある)と比較される。もし完了を待っているならば、命令は完了され、動的活動記録テーブル27中の対応するエントリが消去される。言い換えれば、番号Xのパイプが命令を実行中で、パイプ番号Xが動的活動記録テーブル27の最も古い(一番下の)エントリのパイプ番号欄に記録されている場合、パイプXにおける最も古い命令の完了は、テーブル27中の当該命令に関連するパラメータとパイプXの状況レジスタの内容とを比較することを含む。パイプXの状況レジスタが最も古い命令の実行完了を示していると、その実行結果はパイプXから適切な宛先の方へ送り出される。パイプXの内容は1ステージだけシフトされ、新しい命令を実行のために挿入できるようにする。

次に第9図を参照しながら、Dバス・スタック

29及びDバス・スタック制御回路の動作について説明する。

第9a図において、Dバス・スタック制御回路30のレジスタ74~78は命令用のスタックとして働き、Dバス・スタック29のレジスタ91~93はデータ用のスタックとして働く。Cバス上の命令はDCバス・レジスタ72及びデコーダ73の働きによつてスタックされる。DCバス・レジスタ72は、使用中信号とは無関係に、サイクル毎にCバスをラッチする。デコーダ73は、前の使用中状況の認識に基いて、いつ新しいDバス・ロード命令がFPUに入るかを示す。かくして、データが到着して命令の実行が始まるまで、Dバス・ロード・タイプの各命令に関する情報が維持される。他の重要な状況レジスタは、DRBG有効レジスタ71、S1有効レジスタ80及びS2有効レジスタ79である。これらは何れも2ビットのレジスタであつて、データが部分的に有効か(P)、完全に有効か(F)を示す。レジスタ74~78にある命令が実行されておらず且つ

レジスタ91~93にデータが入っていると、データのオーバーフローが生じる。一方、幾つかの命令がレジスタ74~78に入っているが、レジスタ91~93に十分なデータがなければ、データのアンダーフローが生じる。入力データはDバス・スタック29のレジスタ91~93にスタックされ、入力命令はDバス・スタック制御回路30のレジスタ74~78にスタックされるが、Dバス・データを必要とするスタックされた命令とスタックされたデータとの間には1対1の対応関係がある。Dバスからの一組のデータをパイプライン機構26へ送る場合には、パイプ1~4のうちのどのパイプがこの一組のデータを受取るのかを決定する必要がある。Dバス・スタック29のレジスタ91~93にスタックされたデータと、Dバス・スタック制御回路30のレジスタ78~76にスタックされた命令との間には1対1の対応関係があるので、到着した一組のデータは、Dバス・スタック制御レジスタ74~78のうちの一番下のレジスタ78のビット3及び4(P1P

E NO)によつて表わされるパイプ番号を持つたパイプ(パイプ1~4のうちの1つ)に送られる。

最後に、簡単な命令ストリームの例を示す第12図を参照しながら、この命令ストリームが第1図のハードウェア中をどのように流れるかを説明する。

第12図の命令ストリームは次の3つの命令から成っている。

- (1) FPR1へのロードを行うRXタイプのロード命令。
- (2) FPR1及びFPR2の長精度乗算を行つて、長精度結果をFPR1に書き込むRRタイプの長精度乗算命令。
- (3) FPR3及びFPR4の長精度加算を行つて、長精度結果をFPR3に書き込むRRタイプの長精度加算命令。

この命令ストリームから次のようなことがわかる。

- (1) 各命令はそれぞれ異なつたパイプで実行す

ないことを決定する。この結果、ACK信号(FPU ACK)がハンドシエーク/大域ハザード回路23からパイプ3を初期設定する初期設定回路25へ送られる(複数バス・ユニットPBOの場合はデータ・キャッシュが両ユニットに対して応答するので、FPU ACKは他のユニットには送られない)。初期設定回路25は、ロード命令を動的活動記録テーブル(DHT)27に置くための情報も与える。ロード命令がRXタイプであることがデコーダ73(第9図)で検出されているので、Dバス・スタック制御回路30ではロード命令の情報がDS1レジスタ78に置かれる。また、データ有効信号がDバスからDREG91に受取られているため、DREG有効レジスタ71が活動状態になる。このサイクルの間、CPUはFPUに向けて、RRタイプの乗算命令をCバスへ送り出し、Cバス・レジスタ21b及びDCバス・レジスタ72がこれをラッチする。

サイクル2-RRタイプの乗算命令がデコーダ

る必要がある。

(ロ) ロード命令は、乗算命令で使用されるレジスタ(FPR1)への書き込みを行う。この競合の結果、もし書き込み(ロード)が生じる前に乗算命令が受取られると、インターロックが生じ得る。

(ハ) 加算命令は、前のロード命令及び乗算命令で使用されるFPRを使用しない。

第1図のハードウェアがこの命令ストリームを各サイクルでどのように処理するかを以下に述べる。

サイクル0-CPUがFPUに向けて、RXタイプのロード命令をCバスへ送り出し、Cバス・レジスタ21b(第3図)及びDCバス・レジスタ72(第9図)がこれをラッチする。

サイクル1-ロード命令がデコーダ22及びデコーダ73(第9図)で解読される。ハンドシエーク/大域ハザード回路23が、パイプ3の内部パイプ制御部及び動的活動記録テーブル27を検査することによつて、ロード命令の開始に問題がないこと、及び大域ハザードが存在し

22及び73で解読される。ハンドシエーク/大域ハザード回路23が、パイプ2の内部パイプ制御部及び動的活動記録テーブル27を検査することによつて、乗算の開始に問題がないこと、及び大域ハザードが存在しないことを決定する。この結果、ACK信号(FPU ACK)がCPU及び他のPBUへ送られ、初期設定回路25がパイプ2を初期設定すると共に、FPR2をFPRアレイ28からFLPバスへ読出し、Yレジスタ49にラッチさせる(第6a図)。初期設定回路25は、乗算命令を動的活動記録テーブル27に置くための情報も与える。更に初期設定回路25は、FPR1がロード命令によりインターロックされ、従つて乗算のオペランド1がインターロックされることを、動的活動記録テーブル27を解読したハンドシエーク/大域ハザード回路23から知らせ、デコーダ22からの情報に基づいてFXAレジスタ41及びFYレジスタ44を初期設定する。Dバス・スタック制御回路30のデコーダ73(第9図)

では、乗算命令がDバスを使用しないこと、及びDREG 91によつてインターロックされることを検出しているので、DREG 91がXAレジスタ46(第6a図)にロードされる。DREG 91はこのサイクルで有効であり、ロード命令を完了させるためFPRアレイ28がロード(書込み)される(ロード命令の完了は、次に完了するのがパイプ3であることがDREG有効レジスタ71及び動的活動記録テーブル27により示されると可能である)。CPUはFPUに向けて、RRタイプの加算命令をCバスへ送り出し、Cバス・レジスタ21b及びDCバス・レジスタ72がこれをラッチする。サイクル3-RRタイプの加算命令がデコーダ22及び73で解釈される。ハンドシェーク/大域ハザード回路23が、パイプ1の内部パイプ制御部及び動的活動記録テーブル(DHT)27を検査することによつて、加算の開始に問題がないこと、及び大域ハザードが存在しないことを決定する。この結果、ACK信号(FP

て、整列レジスタ34による整列(アラインメント)もこのサイクルで行われる。

サイクル4-第6a図において、両方のオペランドがXB/3Xレジスタ48及びYレジスタ49で使用可能であり、かくてM1サイクルが開始し、乗算を行うためにM1ハードウェア50が使用される。第5a図において、加算のサイクル2に入り、加算器37が動作して、その出力がSレジスタ38にラッチされる。FAレジスタ31のデータがFSレジスタ33へ転送される。

サイクル5-第6a図において、乗算のM2サイクルが開始してM2ハードウェア51が使用され、その結果がPレジスタ52にラッチされる。FXBレジスタ43がFPレジスタ45をセットする。動的活動記録テーブル(DHT)27は、乗算命令のためのパイプ2が次に完了しなければならないことを示しているので、第5a図のパイプ1は、Sレジスタ38のステージで待機していなければならない。

U ACK)がハンドシェーク/大域ハザード回路23からCPU及び他のPBUへ送られ、初期設定回路25がパイプ1を初期設定すると共に、FPR3及びFPR4をFPRアレイ28からAレジスタ35及びBレジスタ36へ読出す(第5a図)。初期設定回路25は、加算命令を動的活動記録テーブル27に置くための情報も与え、更にインターロックが存在しないことを、動的活動記録テーブル27を解釈したハンドシェーク/大域ハザード回路23から知らせる。また初期設定回路25はデコーダからの情報により、FAレジスタ31及びFBレジスタ32を初期設定する。第6a図において、MUX選択回路53で決定された競合のため、乗算ハードウェアはオペランドをYレジスタ49に保持し、3Xハードウェア47で3倍乗算を実行し、XB/3Xレジスタ48にラッチする。乗算命令のための制御情報はFYレジスタ44に保持され、FXAレジスタ41からFXBレジスタ43へ転送される。第5a図におい

サイクル6-第6a図において、Pレジスタ52が有効であることがFPレジスタ45により示され、且つこのパイプが次に完了することを動的活動記録テーブル(DHT)27が示しているので、乗算命令が完了する。乗算を完了させるため、動的活動記録テーブル27は書込みアドレス及び長さ情報をFPRアレイ28へ供給する。加算はまだSレジスタ38のステージで待機している。

サイクル7-Sレジスタが有効であることがFSレジスタにより示され、且つ加算パイプが次に完了することを動的活動記録テーブル(DHT)27が示しているので、加算が完了する。

これで、第12図の命令ストリームの実行が完了したことになる。

4.図面の簡単な説明

第1図は本発明に従う動的MIMDパイプラインの構成を示すブロック図。

第2図は従来の標準的なMIMDパイプラインの概略を示すブロック図。

第3図は命令スタック21の構成を示すブロック図。

第4図は動的活動記録テーブル27の内容を示す図。

第5a図及び第5b図はパイプライン回路26aの構成を示すブロック図。

第6a図及び第6b図はパイプライン回路26bの構成を示すブロック図。

第7図はパイプライン回路26cの構成を示すブロック図。

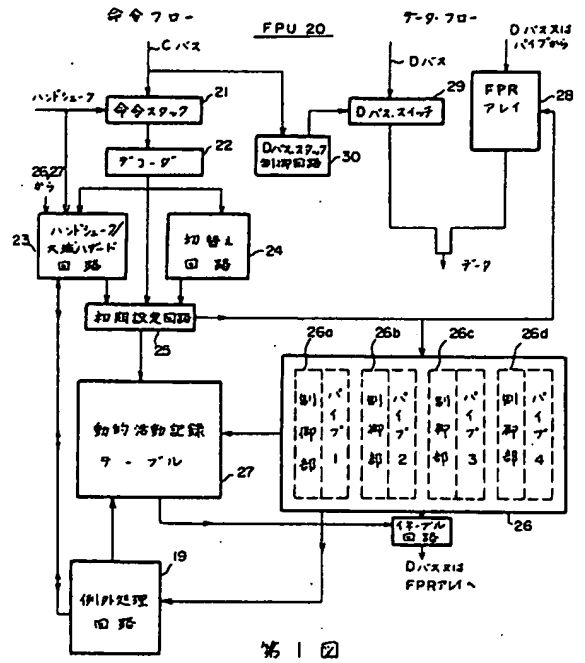
第8図はパイプライン回路26dの構成を示すブロック図。

第9a図及び第9b図はDバス・スタック29及びDバス・スタック制御部30の構成を示すブロック図。

第10図はハンドシェーク／大域ハザード回路23の構成を示すブロック図。

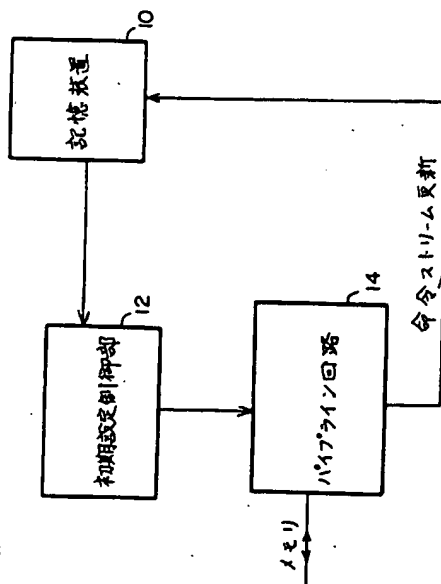
第11図は初期設定回路25の構成を示すブロック図。

第12図は命令ストリームの一例を示す図。



第1図

第2図



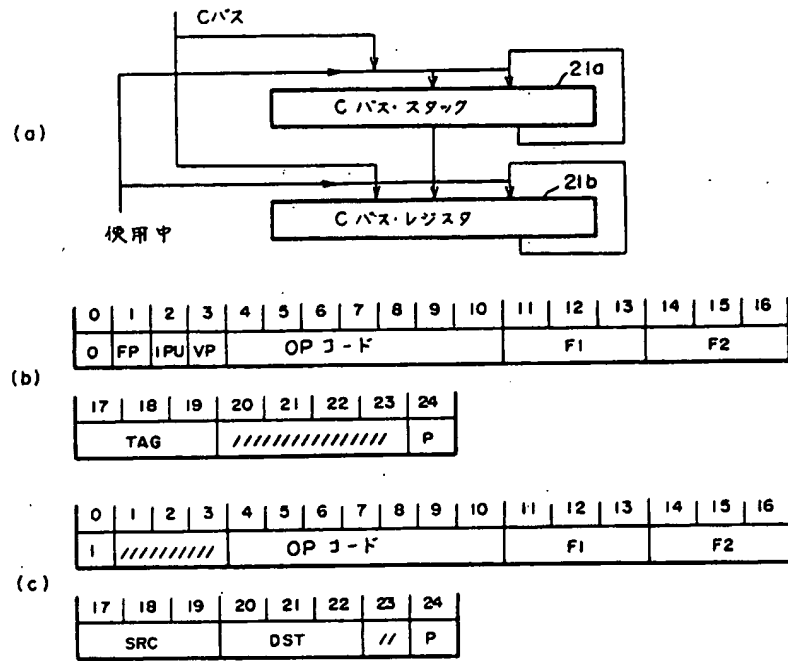
V	WR ADDR			PIPE NO			H	WT	M/S	EXT	LEN	INT TAG					PSW PTR		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16			
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	
	+	+	+									+					+	+	

1
2
3
4
5
6
7
8

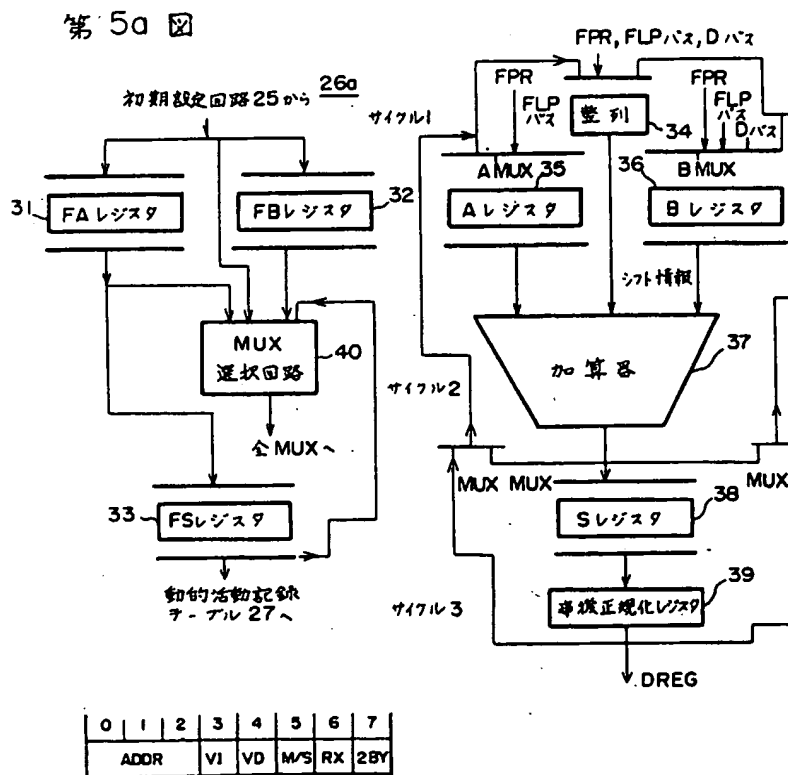
7
27

第 4 页

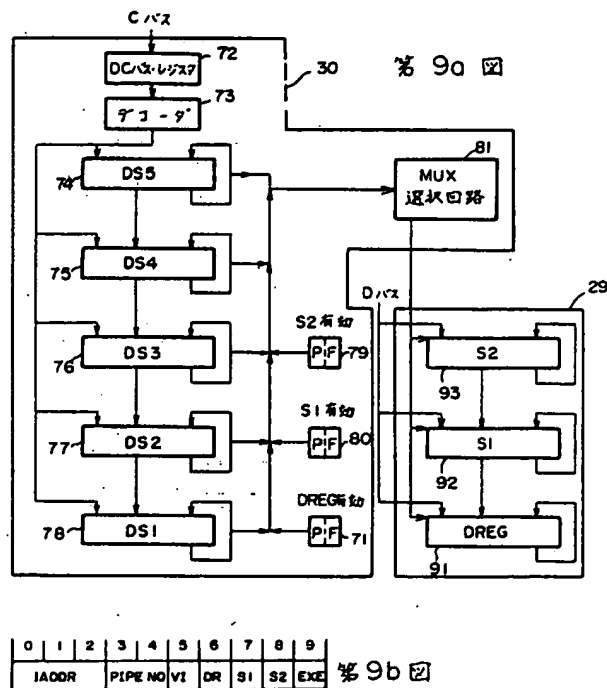
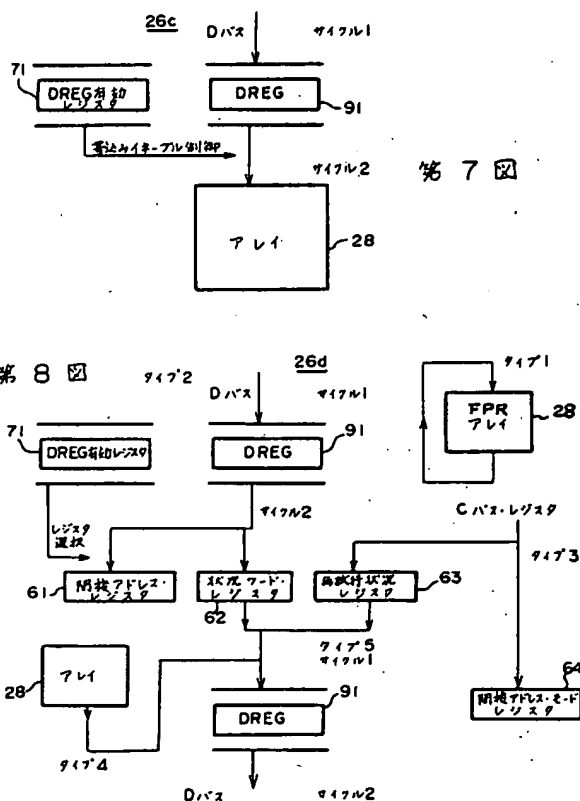
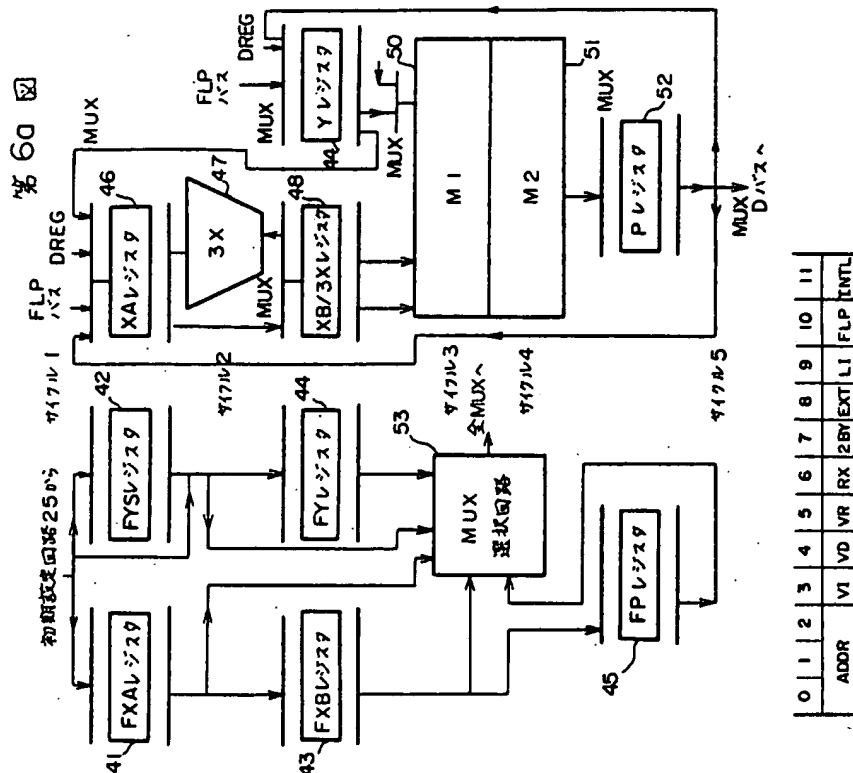
第4図

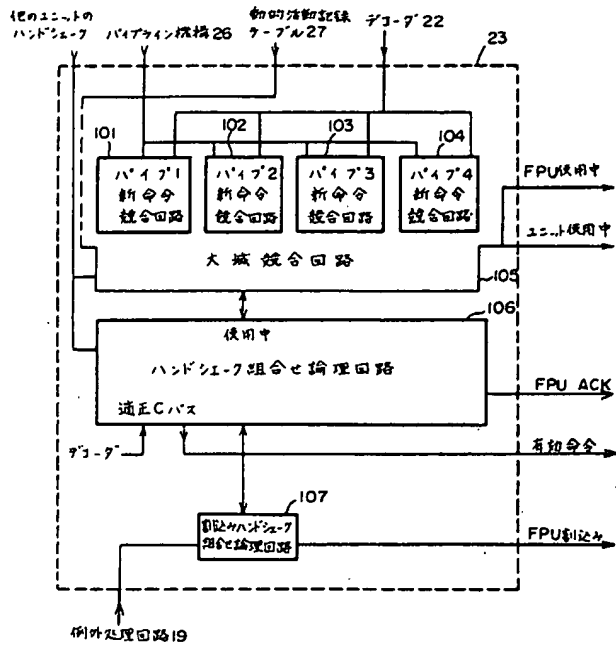


第3図

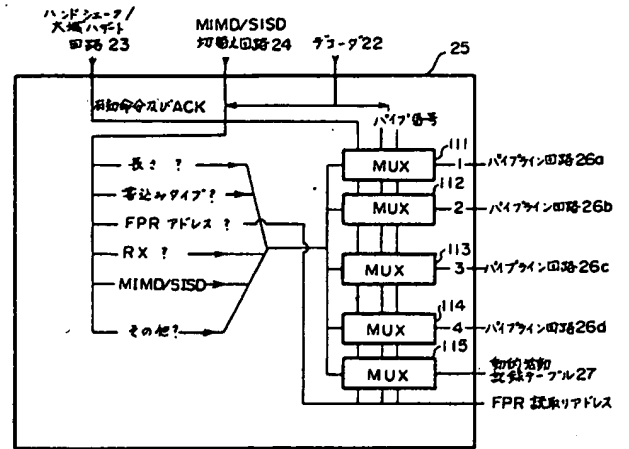


第5b図

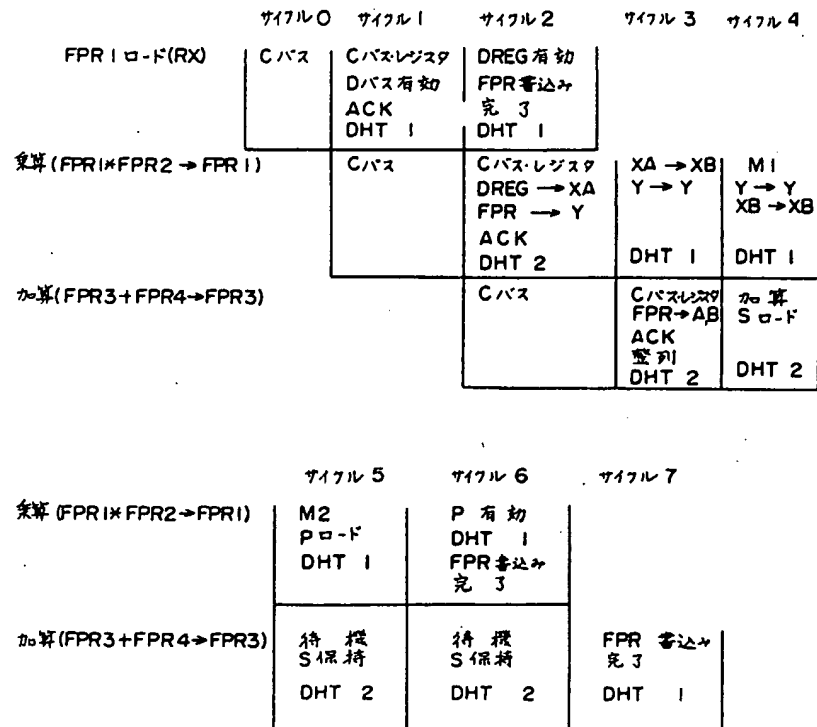




第 10 図



第 11 図



第 12 図